### 📅 ****Day 1: Reverse Engineering the Log Format****

* Analyzed .vlog, .txt, .log samples to understand structure.
* Identified patterns: timestamps, actions (XR-DEL, XR-EXEC), users, and file paths.
* Handled inconsistent formatting (e.g., : vs =>, missing fields).
* Documented field types: timestamp, event type, user,target,optional metadata.

### 📅 ****Day 2: Python Data Parser****

* Developed a Python parser using regex, csv, and datetime.
* Supported both .csv and embedded .vlog formats.
* Handled errors gracefully with try-except blocks.
* Output: parsed data as dictionaries or JSON entries.

### 📅 ****Day 3: Categorization & Timeline****

* Labeled events as user, process, file, or other.
* Sorted by timestamp to build an event timeline.
* Exported to .csv and .json formats.
* Integrated file picker for easy input/output handling.

### 📅 ****Day 4: Anomaly Detection****

* Defined rule-based suspicious sequences (e.g., shadow copy → deletion).
* Detected anomalies per user based on event flow.
* Exported readable anomaly reports with timestamps and raw log lines.

### ****Day 5: Visualizations****

* Used matplotlib and plotly for charts:
* Event frequency
* User activity
* Anomaly flags
* Exported as .png images and interactive .html dashboards.

### ****Day 6: CLI Tool Integration****

* Built a full-featured CLI tool: forensic\_parser.py.
* Supported input directory, format detection, anomaly scanning, and export.
* Command-line options: --dir, --output, --export.
* Ready for extension into GUI or web (e.g., Streamlit).